

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

Claims 1-11 (cancelled)

Claim 12 (new): A method of elliptic curve encryption comprising the step of:

- (a) selecting an elliptic curve $E_p(a,b)$ of the form $y^2 = x^3 + ax + b \pmod{p}$ wherein a and b are non-negative integers less than p satisfying the formula $4a^3 + 27b^2 \pmod{p}$ not equal to 0;
- (b) generating a large 160 bit random number by a method of concatenation of a number of smaller random numbers;
- (c) generating a well hidden point $G(x,y)$ on the elliptic curve $E_p(a,b)$ by scalar multiplication of a point $B(x,y)$ on the elliptic curve with a large random integer which further comprises the steps:
 - (i) converting the large random integer into a series of powers of 2^{31} ;
 - (ii) converting each coefficient of 2^{31} obtained from above step into binary series;
 - (iii) multiplication of binary series obtained from steps (i) and (ii) above with the point $B(x,y)$ on the elliptic curve;
- (d) generating a private key n_A (of about ≥ 160 bit length);
- (e) generating of public key $P_A(x,y)$ given by the formula $P_A(x,y) = (n_A \cdot G(x,y)) \pmod{p}$;
- (f) encrypting the input message MSG;
- (g) decrypting the ciphered text.

Claim 13 (new): The method of elliptic curve encryption as claimed in claim 12, wherein the said number p appearing in selection of elliptic curve is about 160 bit length prime number.

Claim 14 (new): The method of elliptic curve encryption as claimed in claim 12, wherein the said method of generating any large random integer M comprises the steps of:

- (i) setting $I = 0$;
- (ii) setting M to null;
- (iii) determining whether $I < 6$;
- (iv) going to next if true;
- (v) returning M as result if false;
- (vi) generating a random number RI within $(0,1)$;
- (vii) multiplying RI with 10^9 to obtain $BINT$ - an integer of size 9 digits;
- (viii) concatenating $BINT$ to M ;
- (ix) setting $I = I + 1$;
- (x) returning to step (iii).

Claim 15 (new): The method of elliptic curve encryption as claimed in claim 12, wherein said conversion of the large random integer into a series of powers of 2^{31} and conversion of each coefficient m_n of said 2^{31} series thus obtained for scalar multiplication for said random integer with the said point $B(x,y)$ on said elliptic curve $E_p(a,b)$ comprises the steps of:

- (i) accepting a big integer M ;
- (ii) setting $T31$ equal to 2^{31} ;
- (iii) setting $LIM = \text{size of } M \text{ (in bits)}$ and initializing array $A()$ with size LIM ;
- (iv) setting $INCRE$ equal to zero;
- (v) setting N equal to $M \text{ modulus } T31$;
- (vi) setting $M = \text{INT}(M/T31)$;
- (vii) determining whether N is equal to 0;

- (viii) going to next if true;
- (ix) going to step (xxiv) if false;
- (x) determining whether M is equal to 0;
- (xi) going to next if true;
- (xii) going to step (xxvi) if false;
- (xiii) setting $I = 0$ and $J = 0$;
- (xiv) determining whether $I \geq LIM$;
- (xv) going to next step if false;
- (xvi) going to step (xxviii) if true;
- (xvii) determining whether $A(I)$ is equal to 1;
- (xviii) going to next step if true;
- (xix) returning to step (xxii) if false;
- (xx) setting $B(J) = I$;
- (xxi) incrementing J by 1;
- (xxii) incrementing I by 1;
- (xxiii) returning to step (xiv);
- (xxiv) calling function BSERIES (N, INCRE) and updating array A ();
- (xxv) returning to step (x);
- (xxvi) setting $INCRE = INCRE + 31$;
- (xxvii) returning to step (v);
- (xxviii) returning array B () as result.

Claim 16 (new): The method of elliptic curve encryption as claimed in claim 15, wherein said conversion of the large random integer into a series of powers of 2^{31} and said conversion of each coefficient m_n of said 2^{31} series thus obtained for the said scalar multiplication of the said random integer with the said point $B(x,y)$ on said elliptic curve $E_p(a,b)$ further comprises the steps of:

- (i) accepting N and INCRE;

- (ii) assigning BARRAY as an array of values which are powers of $2([2^0, \dots, 2^{30}])$;
- (iii) setting SIZE = size of N (in digits);
- (iv) computing POINTER = 3 (SIZE)+INT(SIZE/3)-4;
- (v) determining whether POINTER < 2;
- (vi) going to next if true;
- (vii) going to step (ix) if false;
- (viii) setting POINTER equal to zero;
- (ix) determining whether (BARRAY(POINTER) \geq N);
- (x) going to next step if true;
- (xi) going to step (xx) if false;
- (xii) determining whether BARRAY (POINTER)=N;
- (xiii) going to next step if true;
- (xiv) going to step (xvii) if false;
- (xv) setting A (POINTER + INCRE) equal to 1;
- (xvi) returning array A () as result;
- (xvii) setting A ((POINTER - 1) + INCRE) equal to 1;
- (xviii) computing N=N-BARRAY(POINTER-1);
- (xix) returning to step (iii);
- (xx) setting POINTER = POINTER + 1;
- (xxi) returning to step (ix).

Claim 17 (new): The method of elliptic curve encryption as claimed in claim 16, wherein said scalar multiplication of the said binary series with the said point B(x,y) on the said elliptic curve $E_p(a,b)$ comprises the steps of:

- (i) accepting B(x,y), a point on $E_p(a,b)$;
- (ii) accepting array B() with size LIM;
- (iii) setting I = 0 and J = 0;
- (iv) determining whether B(J)=I;

- (v) going to next step if true;
- (vi) going to step (xxv) if false;
- (vii) setting $PARR(x,y)$ (J) equal to $B(x,y)$;
- (viii) incrementing J by 1;
- (ix) determining whether J is equal to LIM;
- (x) going to next step if true;
- (xi) going to step (xxv) if false;
- (xii) setting $K=zero$;
- (xiii) determining whether $K>0$;
- (xiv) going to next step if true;
- (xv) going to step (xxii) if false;
- (xvi) computing $FP(x,y)=FP(x,y) + PARR(x,y)$ (K);
- (xvii) incrementing K by 1;
- (xviii) determining whether $K=LIM$;
- (xix) going to next if true;
- (xx) returning to step (xiii) if false;
- (xxi) returning $FP(x,y)$ as result;
- (xxii) setting $FP(x,y)$ equal to $PARR(x,y)$ (K);
- (xxiii) incrementing K by 1;
- (xxiv) returning to step (xiii);
- (xxv) incrementing I by 1;
- (xxvi) setting $B(x,y) = B(x,y) + B(x,y)$;
- (xxvii) returning to step (iv).

Claim 18 (new): The method of elliptic curve encryption as claimed in claim 12, wherein said public key $P_A(x,y)$ is also a point on said elliptic curve $E_p(a,b)$.

Claim 19 (new): The method of elliptic curve encryption as claimed in claim 12, wherein the generation of said private key n_A and said public key $P_A(x,y)$ comprises the steps of:

- (i) entering a big odd integer p of size ≥ 160 bits;
- (ii) determining whether p is a prime number;
- (iii) going to next step if p is prime;
- (iv) going to step (xix) if p is not prime;
- (v) entering a small integer $a > 0$;
- (vi) setting integer $b = 0$ and $x = 1$;
- (vii) determining whether $4a^3 + 27b^2 \bmod (p) = \text{zero}$;
- (viii) going to next step if false;
- (ix) incrementing b by 1 if true and going to step (vii);
- (x) setting $z (=y^2) = x^3 + ax + b$;
- (xi) determining whether $z (=y^2)$ is a perfect square;
- (xii) going to step (xxi) if z is not a perfect square;
- (xiii) setting $B(x,y)$ equal to (x,y) if z is a perfect square;
- (xiv) selecting a large random integer r_1 ;
- (xv) setting $G(x,y) = (r_1 B(x,y)) \bmod (p)$;
- (xvi) selecting a large random integer n_A ;
- (xvii) setting $P_A(x,y) = (n_A \cdot G(x,y)) \bmod (p)$;
- (xviii) return $P_A(x,y)$ as public key and n_A as private key;
- (xix) incrementing p by 2;
- (xx) returning to step (ii);
- (xxi) incrementing x by 1;
- (xxii) determining whether $x > 900$;
- (xxiii) going to next step if true;
- (xxiv) going to step (x) if false;
- (xxv) incrementing b by 1;
- (xxvi) setting $x = 1$;
- (xxvii) returning to step (vii).

Claim 20 (new): The method of elliptic curve encryption as claimed in claim 12, wherein the encryption of said message MSG comprises the steps of:

- (i) generating a large random integer K ;
- (ii) setting $P_{\text{mask}}(x,y) = k \cdot P_A(x,y) \bmod (p)$;
- (iii) setting $P_k(x,y) = k \cdot G(x,y) \bmod (p)$;
- (iv) accepting the message to be encrypted (MSG);
- (v) converting the message into a point $P_c(x,y)$;
- (vi) generating a random point $P_m(x,y)$ on elliptic curve $E_p(a,b)$;
- (vii) setting $P_e(x,y) = (P_c(x,y) - P_m(x,y))$;
- (viii) setting $P_{mk}(x,y) = (P_m(x,y) + P_{\text{mask}}(x,y)) \bmod (p)$;
- (ix) returning $P_k(x)$, $P_e(x,y)$ and $P_{mk}(x)$ as the result (cipher).

Claim 21 (new): The method of elliptic curve encryption as claimed in claim 12, wherein the decryption of said ciphered text comprises the steps of:

- (i) getting cipher text $(P_k(x), P_a(x,y), P_{mk}(x))$;
- (ii) computing $P_k(y)$ from $P_k(x)$ using elliptic curve $E_p(a,b)$;
- (iii) computing $P_{mk}(y)$ from $P_{mk}(x)$ using elliptic curve $E_p(a,b)$;
- (iv) computing $P_{ak}(x,y) = (n_A \cdot P_k(x,y)) \bmod (p)$;
- (v) computing $P_m(x,y) = (P_{mk}(x,y) - P_{ak}(x,y)) \bmod (p)$;
- (vi) computing $P_c(x,y) = (P_m(x,y) + P_e(x,y))$;
- (vii) converting $P_c(x,y)$ into the input message MSG.